I CLAIM:

1. Apparatus for performing a sequence of verification tests to perform hardware and software co-verification on a system under verification, comprising:

a plurality of signal interface controllers operable to be coupled to said system under verification, each signal interface controller being operable to perform one or more test actions transferring at least one of one or more stimulus signals and one or more response signals between a corresponding portion of the system under verification and said signal interface controller during performance of said sequence of verification tests;

a debugger operable to control operation of a processing unit associated with the system under verification, the processing unit being operable to execute software routines;

a debugger signal interface controller operable to interface with the debugger and to perform one or more test actions transferring at least one of one or more stimulus signals and one or more response signals between the debugger and said debugger signal interface controller during performance of said sequence of verification tests; and

a test manager coupled to said plurality of signal interface controllers and the debugger signal interface controller and operable to transfer test controlling messages to said plurality of signal interface controllers and the debugger signal interface controller identifying the test actions to be performed;

the test manager being operable to control the operation of the processing unit via the debugger signal interface controller and the debugger in order to co-ordinate the execution of said software routines with the sequence of verification tests.

2. Apparatus as claimed in Claim 1, further comprising a memory in which the software routines are stored, the debugger signal interface controller being provided with an address indication identifying the addresses of the software routines within the memory, upon receipt of a test controlling message from the test manager, the debugger signal interface controller being operable to generate a corresponding test action with reference to the address indication.

3. Apparatus as claimed in Claim 2, further comprising a status memory operable to store status data, the processing unit being operable to execute monitoring code to monitor the status data in order to identify any changes to the status data and to then execute at least one of said software routines as identified by the change in status data, the corresponding test action being arranged to cause updated status data identifying a corresponding one of said software routines to be stored in the status memory under the control of the debugger.

4. Apparatus as claimed in Claim 3, wherein the debugger is operable to cause the processing unit to store the updated status data in the status memory, whereafter the processing unit reverts to executing the monitoring code.

5. Apparatus as claimed in Claim 2, wherein the processing unit has a plurality of registers associated therewith, and the corresponding test action is arranged to cause data in a selected register of said plurality of registers to be updated under the control of the debugger, such that the processing unit will then execute a corresponding one of said software routines.

6. Apparatus as claimed in Claim 5, wherein the selected register is operable to store a program counter value and the corresponding test action is arranged to cause the program counter value to be updated in order to cause the processing unit to branch to the corresponding one of said software routines.

7. Apparatus as claimed in Claim 1, wherein upon execution of one of said software routines, the processing unit is operable to update status data indicative of whether the software routine completed successfully, the debugger signal interface controller being operable to perform a predetermined test action in order to cause a breakpoint to be set by the debugger which is triggered when the processing unit performs said update of the status data.

8. Apparatus as claimed in Claim 7, wherein the debugger is operable to issue a callback event to the debugger signal interface controller upon triggering of the breakpoint.

9. Apparatus as claimed in Claim 1, wherein said stimulus and/or response signals are transferred between the debugger signal interface controller and the debugger using Application Programming Interface (API) calls.

10. Apparatus as claimed in Claim 2, wherein at least one of the software routines is written into the memory via the debugger under the control of the debugger signal interface controller.

11. Apparatus as claimed in Claim 1, wherein multiple processing units are provided, and a corresponding multiple of debugger signal interface controllers are provided, each debugger signal interface controller communicating with the same debugger to cause their respective test actions to be performed.

12. Apparatus as claimed in Claim 1, wherein the timing of the execution of said software routines is co-ordinated with the sequence of verification tests.

13. Apparatus as claimed in Claim 1, wherein the system under verification comprises a plurality of components, each signal interface controller being associated with one of said components.

14. Apparatus as claimed in Claim 13, wherein the processing unit forms one of the components of the system under verification.

15. Apparatus as claimed in Claim 1, further comprising the processing unit, the processing unit being provided externally to the system under verification.

16. Apparatus as claimed in Claim 1, wherein the processing unit comprises a representation of a processor on which the software routines are intended to be executed.

17. Apparatus as claimed in Claim 1, wherein the system under verification comprises a hardware simulator responsive to said one or more stimulus signals to generate said one or more response signals simulating a response of a data processing apparatus to said one or more stimulus signals if applied to said data processing apparatus.

18. A method of performing a sequence of verification tests to perform hardware and software co-verification on a system under verification, comprising the steps of:

performing in each of a plurality of signal interface controllers coupled to said system under verification one or more test actions transferring at least one of one or more stimulus signals and one or more response signals between a corresponding portion of the system under verification and said signal interface controller during performance of said sequence of verification tests;

controlling via a debugger execution of software routines by a processing unit associated with the system under verification;

performing in a debugger signal interface controller coupled with the debugger one or more test actions transferring at least one of one or more stimulus signals and one or more response signals between the debugger and said debugger signal interface controller during performance of said sequence of verification tests; and

transferring test controlling messages from a test manager to said plurality of signal interface controllers and to the debugger signal interface controller, the test controlling messages identifying the test actions to be performed;

whereby the test manager controls the operation of the processing unit via the debugger signal interface controller and the debugger in order to co-ordinate the execution of said software routines with the sequence of verification tests.

19. A method as claimed in Claim 18, further comprising the steps of:

storing the software routines in a memory;

providing the debugger signal interface controller with an address indication identifying the addresses of the software routines within the memory; and

upon receipt of a test controlling message from the test manager, generating in the debugger signal interface controller a corresponding test action with reference to the address indication.

20. A method as claimed in Claim 19, further comprising the steps of:

storing status data in a status memory;

executing on the processing unit monitoring code to monitor the status data in order to identify any changes to the status data and then executing at least one of said software routines as identified by the change in status data, the corresponding test action causing updated status data identifying a corresponding one of said software routines to be stored in the status memory under the control of the debugger.

21. A method as claimed in Claim 20, further comprising the step of:

causing the processing unit to store the updated status data in the status memory, whereafter the processing unit reverts to executing the monitoring code.

22. A method as claimed in Claim 19, wherein the processing unit has a plurality of registers associated therewith, and the corresponding test action causes data in a selected register of said plurality of registers to be updated under the control of the debugger, such that the processing unit will then execute a corresponding one of said software routines.

23. A method as claimed in Claim 22, wherein the selected register is operable to store a program counter value and the corresponding test action causes the program counter value to be updated in order to cause the processing unit to branch to the corresponding one of said software routines.

24. A method as claimed in Claim 18, further comprising the steps of:

upon execution of one of said software routines, employing the processing unit to update status data indicative of whether the software routine completed successfully; and

causing the debugger signal interface controller to perform a predetermined test action in order to cause a breakpoint to be set by the debugger which is triggered when the processing unit performs said update of the status data.

25. A method as claimed in Claim 24, wherein the debugger issues a callback event to the debugger signal interface controller upon triggering of the breakpoint.

26. A method as claimed in Claim 18, wherein said stimulus and/or response signals are transferred between the debugger signal interface controller and the debugger using Application Programming Interface (API) calls.

27. A method as claimed in Claim 19, further comprising the step of:

writing at least one of the software routines into the memory via the debugger under the control of the debugger signal interface controller.

28. A method as claimed in Claim 18, wherein multiple processing units are provided, and a corresponding multiple of debugger signal interface controllers are provided, each debugger signal interface controller communicating with the same debugger to cause their respective test actions to be performed.

29. A method as claimed in Claim 18, wherein the timing of the execution of said software routines is co-ordinated with the sequence of verification tests.

30. A method as claimed in Claim 18, wherein the system under verification comprises a plurality of components, each signal interface controller being associated with one of said components.

31. A method as claimed in Claim 30, wherein the processing unit forms one of the components of the system under verification.

32. A method as claimed in Claim 18, wherein the processing unit is provided externally to the system under verification.

33. A method as claimed in Claim 18, wherein the processing unit comprises a representation of a processor on which the software routines are intended to be executed.

34. A method as claimed in Claim 18, wherein the system under verification comprises a hardware simulator which in response to said one or more stimulus signals generates said one or more response signals simulating a response of a data processing apparatus to said one or more stimulus signals if applied to said data processing apparatus.

35. A computer program product for performing a sequence of verification tests to perform hardware and software co-verification on a system under verification, the computer program product comprising:

a plurality of signal interface controller code blocks operable to be coupled to said system under verification, each signal interface controller code block being operable to perform one or more test actions transferring at least one of one or more stimulus signals and one or more response signals between a corresponding portion of the system under verification and said signal interface controller code block during performance of said sequence of verification tests;

debugger code operable to control operation of a processing unit associated with the system under verification, the processing unit being operable to execute software routines;

a debugger signal interface controller code block operable to interface with the debugger code and to perform one or more test actions transferring at least one of one or more stimulus signals and one or more response signals between the debugger code and

said debugger signal interface controller code block during performance of said sequence of verification tests; and

test manager code coupled to said plurality of signal interface controller code blocks and the debugger signal interface controller code block and operable to transfer test controlling messages to said plurality of signal interface controller code blocks and the debugger signal interface controller code block identifying the test actions to be performed;

the test manager code being operable to control the operation of the processing unit via the debugger signal interface controller code block and the debugger code in order to co-ordinate the execution of said software routines with the sequence of verification tests.

36. A computer program product as claimed in Claim 35, wherein the software routines are stored in a memory, the debugger signal interface controller code block being provided with an address indication identifying the addresses of the software routines within the memory, upon receipt of a test controlling message from the test manager code, the debugger signal interface controller code block being operable to generate a corresponding test action with reference to the address indication.

37. A computer program product as claimed in Claim 36, wherein status data is stored in a status memory, the processing unit being operable to execute monitoring code to monitor the status data in order to identify any changes to the status data and to then execute at least one of said software routines as identified by the change in status data, the corresponding test action being arranged to cause updated status data identifying a corresponding one of said software routines to be stored in the status memory under the control of the debugger code.

38. A computer program product as claimed in Claim 37, wherein the debugger code is operable to cause the processing unit to store the updated status data in the status memory, whereafter the processing unit reverts to executing the monitoring code.

39. A computer program product as claimed in Claim 36, wherein the processing unit has a plurality of registers associated therewith, and the corresponding test action is arranged to cause data in a selected register of said plurality of registers to be updated under the control of the debugger code, such that the processing unit will then execute a corresponding one of said software routines.

40. A computer program product as claimed in Claim 39, wherein the selected register is operable to store a program counter value and the corresponding test action is arranged to cause the program counter value to be updated in order to cause the processing unit to branch to the corresponding one of said software routines.

41. A computer program product as claimed in Claim 35, wherein upon execution of one of said software routines, the processing unit is operable to update status data indicative of whether the software routine completed successfully, the debugger signal interface controller code block being operable to perform a predetermined test action in order to cause a breakpoint to be set by the debugger code which is triggered when the processing unit performs said update of the status data.

42. A computer program product as claimed in Claim 41, wherein the debugger code is operable to issue a callback event to the debugger signal interface controller code block upon triggering of the breakpoint.

43. A computer program product as claimed in Claim 35, wherein said stimulus and/or response signals are transferred between the debugger signal interface controller code block and the debugger code using Application Programming Interface (API) calls.

44. A computer program product as claimed in Claim 36, wherein at least one of the software routines is written into the memory via the debugger code under the control of the debugger signal interface controller code block.

45. A computer program product as claimed in Claim 35, wherein multiple processing units are provided, and a corresponding multiple of debugger signal interface controller code blocks are provided, each debugger signal interface controller code block communicating with the same debugger code to cause their respective test actions to be performed.

46. A computer program product as claimed in Claim 35, wherein the timing of the execution of said software routines is co-ordinated with the sequence of verification tests.

47. A computer program product as claimed in Claim 35, wherein the system under verification comprises a plurality of components, each signal interface controller code block being associated with one of said components.

48. A computer program product as claimed in Claim 47, wherein the processing unit forms one of the components of the system under verification.

49. A computer program product as claimed in Claim 35, wherein the processing unit is provided externally to the system under verification.

50. A computer program product as claimed in Claim 35, wherein the processing unit comprises a representation of a processor on which the software routines are intended to be executed.

51. A computer program product as claimed in Claim 35, wherein the system under verification comprises hardware simulator code responsive to said one or more stimulus signals to generate said one or more response signals simulating a response of a data processing apparatus to said one or more stimulus signals if applied to said data processing apparatus.